PRAC4: ICA: Independent Component Analysis

Asignatura: CLP: Clasificación de Patrones.
Optativa de 2º Ciclo
ETSETB
UPC

Profesores:
Margarita Cabrera
Josep Vidal
UPC-TSC-D5

Abril-2007

1 Introducción

Objetivos de la práctica:

- Se va a comparar la separación ciega de fuente sobre señales de audio aplicando técnicas ICA y técnicas PCA
- Se van a comparar las técnicas PCA, MDA y ICA como etapas de pre-procesado en clasificación de señales.

La base de datos “Voces” ha sido facilitada por el profesor Antonio Bonafonte (TALP, UPC, http://gps-tsc.upc.es/veu/personal/antonio).

El software: http://www.cis.hut.fi/projects/ica/fastica/ se utiliza en esta práctica con autorización por parte de los autores .

Además se utilizará la base de datos SPAM y la base de datos FONEMAS (Práctica 2) y el software prtools para el clasificador ldc y para el qdc propuesto en el apartado 3.2.

1.1 Características de la base de datos: Voces

Se tiene el siguiente conjunto de ficheros de audio. Todos han sido grabados a la frecuencia de muestreo de 8000 KHz y 8 bits por muestra.

Una mujer, párrafos, en catalán.
==========
camapa0070 (105727 muestras)
camapa0071 (114175 muestras)
camapa0072 (149503 muestras)

La misma frases, mujer, en castellano
==========
f1fr420 (24064 muestras)
f1fr421 (29184 muestras)
f1fr422 (24064 muestras)
f1fr423 (18944 muestras)
f1fr424 (29440 muestras)
f1fr425 (24576 muestras)
f1fr426 (26368 muestras)
f1fr427 (25088 muestras)
f1fr428 (30976 muestras)
f1fr429 (25600 muestras)

Las mismas frases, hombre, en castellano
==========
m1fr420 (23808 muestras)
m1fr421 (33024 muestras)
m1fr422 (25856 muestras)
m1fr423 (22528 muestras)
m1fr424 (29696 muestras)
m1fr425 (25344 muestras)
m1fr426 (30464 muestras)
m1fr427 (27136 muestras)
m1fr428 (26624 muestras)
m1fr429 (26112 muestras)

Frases de una mujer mexicana
==========
mxse0480 (85248 muestras)
mxse0481 (82432 muestras)
mxse0482 (82688 muestras)
mxse0483 (79872 muestras)
mxse0484 (85760 muestras)

1.2 Software Fastica

The FastICA package is a free (GPL) MATLAB program that implements the fast fixed-point algorithm for independent component analysis and projection pursuit. It features an easy-to-use graphical user interface, and a computationally powerful algorithm.

Este paquete de software de Matlab es un interfaz gráfico que a partir de la mezcla se señales, realiza:

1º Preblanqueado de la señal mediante la técnica PCA (Principal Component Analysis)
2º Sobre las señales resultantes aplica ICA: Independent Component Análisis

El algoritmo utilizado es de "punto fijo" para la obtención de las diferentes componentes independientes.
Maximiza la función Kurtosis
Se puede elegir entre 4 tipos distintos de no linealidades para la función g(.) (Derivada de la función G(.) que aproxima la Kurtosis por:

$$J(y) \approx \left[E\{G(y)\} - E\{G(\upsilon)\} \right]^2$$

Ecuación implementada:

1.- Valor aleatorio inicial para $\mathbf{w}[n]$

2.- $\mathbf{w}[n+1] = E\left\{\mathbf{z}(t) g\left(\mathbf{w}^T[n]\mathbf{z}(t)\right)\right\} - E\left\{g'\left(\mathbf{w}^T[n]\mathbf{z}(t)\right)\mathbf{w}[n]\right\}$

3.- $\mathbf{w}[n+1] = \mathbf{w}[n+1] / \left\|\mathbf{w}[n+1]\right\|$

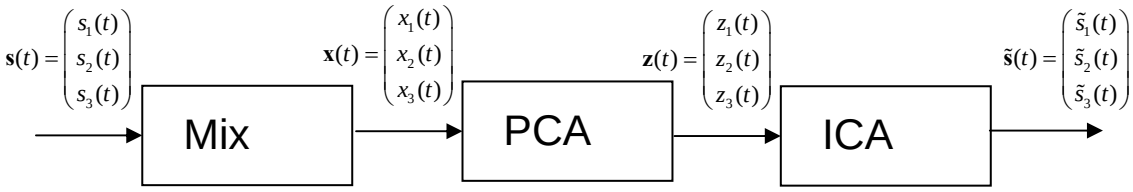
4.- Si no converge volver a 2

No linealidades:

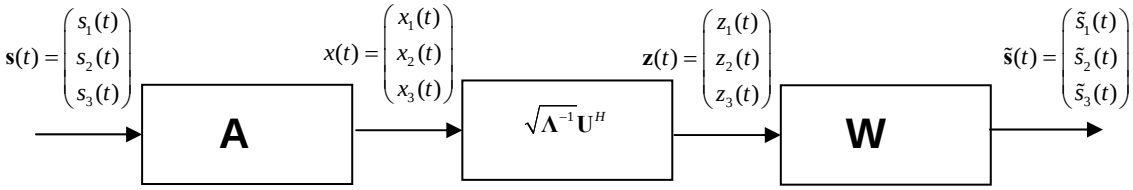
Pow3:	$g(y) = y^3$
Tanh:	$g(y) = \text{th}(ay) = \frac{e^{ay} - e^{-ay}}{e^{ay} + e^{-ay}}$
Gauss:	$g(y) = y \exp\left(-a \frac{y^2}{2}\right)$
Skew:	$g(y) = y^2$

2 Escenario de Trabajo

Mezclando señales de audio de la base de datos Voces y/o señales de ruido se simulará el problema popularmente conocido como “Cocktail Party”:



En forma matricial:



Señales originales independientes (Sig_Source): $\mathbf{s}(t)=\begin{pmatrix}s_1(t)\\s_2(t)\\s_3(t)\end{pmatrix}$

Señales Mezcladas (Sig_Mixed): $\mathbf{x}(t)=\begin{pmatrix}x_1(t)\\x_2(t)\\x_3(t)\end{pmatrix}$; $\mathbf{x}(t)=\mathbf{A}\mathbf{s}(t)$

Señales Blanqueadas: (Sig_PCA):

$$\mathbf{z}(t)=\begin{pmatrix}z_1(t)\\z_2(t)\\z_3(t)\end{pmatrix};\quad \mathbf{z}(t)=\sqrt{\mathbf{\Lambda}^{-1}}\mathbf{U}^H\mathbf{x}(t)=\sqrt{\mathbf{\Lambda}^{-1}}\mathbf{U}^H\mathbf{A}\mathbf{s}(t)=\mathbf{P}_{PCA}\mathbf{s}(t)$$

Señales Independientes:(Sig_ICA):

$$\tilde{\mathbf{s}}(t)=\begin{pmatrix}\tilde{s}_1(t)\\\tilde{s}_2(t)\\\tilde{s}_3(t)\end{pmatrix};\quad \tilde{\mathbf{s}}(t)=\mathbf{W}\mathbf{z}(t)=\mathbf{W}\sqrt{\mathbf{\Lambda}^{-1}}\mathbf{U}^H\mathbf{x}(t)=\mathbf{W}\sqrt{\mathbf{\Lambda}^{-1}}\mathbf{U}^H\mathbf{A}\mathbf{s}(t)=\mathbf{P}_{ICA}\mathbf{s}(t)$$

Para aplicar la técnica PCA (Whitening), en el programa facilitado, se aplica previa substracción de la media temporal, la descomposición svd de la matriz de mezcla A.

- Cuando la matriz A se halla disponible en el procesador, situación NO habitual en la práctica, se puede calcular la descomposición en valores singulares de dicha matriz (utilizando por ejemplo *svd.m*):

$$svd(\mathbf{A}) = \mathbf{U}\sqrt{\mathbf{\Lambda}}\mathbf{V}^H; \quad \mathbf{U}\mathbf{U}^H = \mathbf{I}; \quad \mathbf{V}\mathbf{V}^H = \mathbf{I}; \quad \sqrt{\mathbf{\Lambda}} : diagonal$$

- Cuando la matriz A no se halla disponible en el procesador, situación habitual en la práctica, se puede calcular la matriz de blanqueado W_PCA= $\sqrt{\mathbf{\Lambda}^{-1}}\mathbf{U}$, a partir de la descomposición en autovectores y autovalores principales de la matriz de covarianza (utilizando por ejemplo *eig.m*):

$$\mathbf{C} = \mathrm{E}\left[\mathbf{x}(t)\mathbf{x}^T(t)\right] = \mathrm{E}\left[\mathbf{A}\mathbf{s}(t)\mathbf{s}^T(t)\mathbf{A}^T\right] = \mathbf{A}\mathrm{diag}(P_i)\mathbf{A}^T = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^H$$

Para Medir la semejanza entre las señales fuente (Sig_Source) y las señales separadas (Sig_PCA, Sig_ICA), se propone que:

- Normalice todas las señales (Sig_Source, Sig_PCA y Sig_ICA) por sus correspondientes coeficientes de correlación.

$$(\rho_i)^2 = \frac{\mathrm{E}\left[s_i(t)s_i(t)\right]}{\sqrt{\mathrm{E}\left[\left\|s_i(t)\right\|^2\right]\mathrm{E}\left[\left\|s_i(t)\right\|^2\right]}}.$$

Y calcule la forma de la matriz de permutación $\mathbf{P}_{PCA}, \mathbf{P}_{ICA}$ de la página 5.

Cuando se dispone de la matriz de mezcla **A**:

- Averigüe la forma de la matriz de permutación más aproximada:

$$\mathbf{P}_{PCA} \simeq \sqrt{\mathbf{\Lambda}^{-1}}\mathbf{U}\mathbf{A} = W\_PCA * A \qquad \mathbf{P}_{ICA} \simeq \mathbf{W}\sqrt{\mathbf{\Lambda}}\mathbf{U}\mathbf{A} = W\_ICA * A$$

$$\tilde{\mathbf{s}}(t) = \begin{pmatrix} \tilde{s}_1(t) \\ \tilde{s}_2(t) \\ \tilde{s}_3(t) \end{pmatrix}; \quad \mathbf{s}(t) = \begin{pmatrix} s_1(t) \\ s_2(t) \\ s_3(t) \end{pmatrix}$$

$$\tilde{\mathbf{s}}_{PCA}(t) = \mathbf{z}(t) = \sqrt{\mathbf{\Lambda}^{-1}}\mathbf{U}^H\mathbf{A}\mathbf{s}(t) \simeq \mathbf{P}_{PCA}\ \mathbf{s}(t)$$
$$\tilde{\mathbf{s}}_{ICA}(t) = \tilde{\mathbf{s}}(t) = \mathbf{W}\sqrt{\mathbf{\Lambda}^{-1}}\mathbf{U}^H\mathbf{A}\mathbf{s}(t) \simeq \mathbf{P}_{ICA}\ \mathbf{s}(t)$$

Cuando NO se dispone de la matriz de mezcla **A**:

- Averigüe la forma de la matriz de permutación más aproximada:

$$\mathbf{P}_{PCA} \simeq Sig\_PCA * Sig\_source^{H} \qquad \mathbf{P}_{ICA} \simeq Sig\_ICA * Sig\_source^{H}$$

En las estimaciones de matriz de permutación anteriores, normalice previamente las señales recuperadas tanto en Sig_PCA como en Sig_source para que tengan potencia igual a la unidad. Las señales de Sig_source se normalizan ya dentro del fichero "prac4_ICA.m".

La matriz de permutación estimada, la puede también normalizar por el máximo valor de la misma con el objeto de interpretar mejor los resultados.

Nota: Matriz de Permutación P: Matriz cuadrada con todos sus n×n elementos iguales a 0, excepto uno cualesquiera por cada fila y columna, el cual debe ser igual a 1.

Ejemplo para n = 3 : $\mathbf{P} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$

Permutación de Filas: **PA** envía la primera fila a la tercera, la segunda a la primera y la tercera a la segunda.

Permutación de Columnas: **AP** envía la primera columna a la tercera, la segunda a la primera y la tercera a la segunda.

3 Laboratorio

3.1 PCA e ICA en Separación Ciega de Fuentes

1.- Lea el contenido del fichero prac4_ICA.m e identifique las distintas etapas con los bloques de la figura presentada.
2.- Realice una ejecución del mismo y evalúe el resultado de forma subjetiva a partir de las gráficas obtenidas y de las audiciones de todas las señales involucradas.
3.- Programe el cálculo de la matriz de blanqueado W_PCA= $\sqrt{\mathbf{\Lambda}^{-1}}\mathbf{U}$, según se indica en el apartado anterior a partir de la matriz de covarianza de Sig_Mixed.

Trabajando con 3 señales de mezcla mida las matrices de permutación PCA y ICA: $\mathbf{P}_{PCA}, \mathbf{P}_{ICA}$.
Pruébelo para las siguientes situaciones y compruebe la lógica de los resultados obtenidos. Tenga en cuenta que una matriz de mezcla perfectamente ortogonal es una situación poco realista.

- 3 Señales de Audio, Matriz de Mezcla Ortogonal y matriz W_PCA estimada a partir de la matriz A.
- 3 Señales de Audio, Matriz de Mezcla Ortogonal y matriz W_PCA estimada a partir de la matriz de covarianza de Sig_Mixed.
- 3 Señales de Audio, Matriz de Mezcla NO Ortogonal y matriz W_PCA estimada a partir de la matriz A.
- 3 Señales de Audio, Matriz de Mezcla NO Ortogonal y matriz W_PCA estimada a partir de la matriz de covarianza de Sig_Mixed.

- 1 Señal de Audio y dos señales de ruido gaussiano, Matriz de Mezcla Ortogonal y matriz W_PCA estimada a partir de la matriz de covarianza de Sig_Mixed.
- 1 Señal de Audio y dos señales de ruido gaussiano, Matriz de Mezcla NO Ortogonal y matriz W_PCA estimada a partir de la matriz de covarianza de Sig_Mixed.
- 1 Señal de Audio y dos señales de ruido Laplaciano (Prac0_P05), Matriz de Mezcla NO Ortogonal y matriz W_PCA estimada a partir de la matriz de covarianza de Sig_Mixed.

Guarde los resultados de correlación anteriores (es decir, las matrices de permutación) en una única tabla Word.

3.2 PCA e ICA en Clasificación

Modifique el fichero prac2_spam_PCA.m. Utilizando el criterio de clasificación *ldc*, y el de *qdc*, mida el error de clasificación en las siguientes situaciones:

- Aplicando reducción de coordenadas mediante PCA: Nueva dimensión. d=1:1:57. Dibuje el error de clasificación en función de d (Esta parte debería estar ya realizada en la práctica 2).
- Aplicando reducción de coordenadas mediante ICA: d=1:1:57 (Si va muy lento realice los saltos de 5 en 5). Dibuje el error de clasificación en función de d.

- Aplicando reducción de coordenadas mediante MDA: dnew=c-1=1 (Única posibilidad ya que solo hay dos clases).

Repita los resultados anteriores con la base de datos FONEMAS (d=64), trabajando con las 5 clases. (prac2_fonemas.m) Observe que en MDA las nuevas dimensiones varían según dnew =1:c-1; c=5;

COMENTE TODOS LOS RESULTADOS EN EL DOCUMENTO:

PRAC4 _G*.doc**

4 Software a utilizar

4.1 Programa prac4_ICA.m

```
%This program:
%   1- Reads Audio signals and generates noise signals
%   2- Mix the Signals with a square full-rank matrix A that can be
%   orthogonal or non orthogonal
%   3- Whites the Signals
%   4- Applies ICA to the Mixed Signals


%   INPUT Parameters
%*****************************************************************

N_Voces=3;                  %Numero de señales de Audio
N_Ruido=0;                  %Numero de señales de Ruido
N_Signal=N_Voces+N_Ruido;
i_UnitaryMatrix=input('Orthogonal(1),Non orth.<>1 = ');                 %1(Orthogonal
Unitary Matrix)

% Length of Signals is limited to avoid memory problems:

N=10000;                    %Longitud maxima de señales a procesar
N_tran=2000;                %Transitorio, recomendable para eliminar silencios
                            %al principio de los ficheros de audio,
                            % N=N-N_tran;

%*************************************************************
%   1- Reads Audio signals and generates noise signals
%*************************************************************
% Audio Reading
NAME1='.wav';
Sig_source=zeros(1,N);
for i_Voces=1:N_Voces
```

```
   NAME=input('File Name: ','s');  %Nombre ficheros
   T= strcat(NAME,NAME1);
   [s1,FS,NBITS]=wavread(T);
   N=min([length(s1) N]);
   Sig_source=[Sig_source(:,1:N);s1(1:N)'];
end
Sig_source=Sig_source(2:N_Voces+1,N_tran+1:N);
N=N-N_tran;
%Noise Generation
for i_Voces=1:N_Ruido
   Sig_source=[Sig_source;randn(1,N)];
end

Sig_source=Sig_source-mean(Sig_source')'*ones(1,N);

for i_signal=1:N_Signal
   s1=Sig_source(i_signal,:);
   s1=s1/sqrt(s1*s1');
   Sig_source(i_signal,:)=s1;
end

eje_t=(1:N)/FS;

%PLOTSSS
 figure('name','Sources')
for i1=1:N_Signal
   subplot(N_Signal,1,i1)
   plot(eje_t,Sig_source(i1,:))
   grid
   Sig_Aux=Sig_source(i1,:)/(2*max(abs(Sig_source(i1,:))));
   sound(Sig_Aux,FS)
   pause
end

%*****************************************************************
%   2- Mix the Signals with a square full-rank matrix A that can be
%   orthogonal or non orthogonal
%*****************************************************************
%Mixing Matrix
A=randn(N_Signal,N_Signal);
if i_UnitaryMatrix==1
   A=orth(A);
   D1=diag([1:-0.1:1-0.1*(N_Signal-1)]);
   A=A*D1;
end

% Mixing Signals;
Sig_Mixed=A*Sig_source;

%PLOTSSS
```

```
 figure('name','MIXED')
for i1=1:N_Signal
   subplot(N_Signal,1,i1)
   plot(eje_t,Sig_Mixed(i1,:))
   grid
   Sig_Aux=Sig_Mixed(i1,:)/(2*max(abs(Sig_Mixed(i1,:))));
   sound(Sig_Aux,FS)
   pause
end

%*****************************************************
%   3- Whites the Signals: PCA applied to Mixed Signal
%*****************************************************

[U,S,V] = svd(A);       % A=U*S*V'
W_PCA=inv(S)*U';

Sig_PCA=W_PCA*Sig_Mixed;

%PLOTSSS
figure('name','WHITED')
for i1=1:N_Signal
   subplot(N_Signal,1,i1)
   plot(eje_t,Sig_PCA(i1,:))
   grid
   Sig_Aux=Sig_PCA(i1,:)/(2*max(abs(Sig_PCA(i1,:))));
   sound(Sig_Aux,FS)
   pause
end

%*****************************************************
%   4- Applies ICA to the Mixed Signals
%*****************************************************
 [Sig_ICA, A_ICA, W_ICA] = fastica(Sig_PCA);

%PLOTSSS
figure('name','ICA')
for i1=1:N_Signal
   subplot(N_Signal,1,i1)
   plot(eje_t,Sig_ICA(i1,:))
   grid
   Sig_Aux=Sig_ICA(i1,:)/(2*max(abs(Sig_ICA(i1,:))));
   sound(Sig_Aux,FS)
   pause
end
```

4.2 Método Gráfico Alternativo para separación de Fuentes mediante ICA

Este método se puede utilizar como alternativo al proporcionado en 4.1. Puede ayudarle a entender mejor el software utilizado para ICA. Se recomienda que lo pruebe como opcional, al final de la práctica.

1.- Ejecutar Prac4_ICA.m
Únicamente para preparar las señales de entrada.

2.- Ejecutar “fasticag” que es la aplicación gráfica, Aplicar PCA e ICA y guardar todas las señales. Pedira el nombre de las variables.

3.- Ejecutar prac4_results.m para visualizar y escuchar los resultados.

4.3 Reducción de dimensión por PCA, ICA, MDA

Ejemplo de reducción de dimensión con criterio PCA:

```
[W_pca,alf] = pca(A_total,i_new);
A_2=A_total*W_pca;
```

Donde A_total ya es una base de datos formada con software prtools

Ejemplo de reducción de dimensión con criterio MDA:

```
[W_mda] = fisherm(A_total,i_new);
A_2=A_total*W_mda;
```

Donde A_total ya es una base de datos formada con software prtools

Ejemplo de reducción de dimensión con criterio ICA:

```
[icasig] = fastica (signal', 'lastEig', i_new);
A_2=dataset(icasig',labs);
```

Donde signal es la matriz correspondiente a la base de datos y labs sus etiquetas;